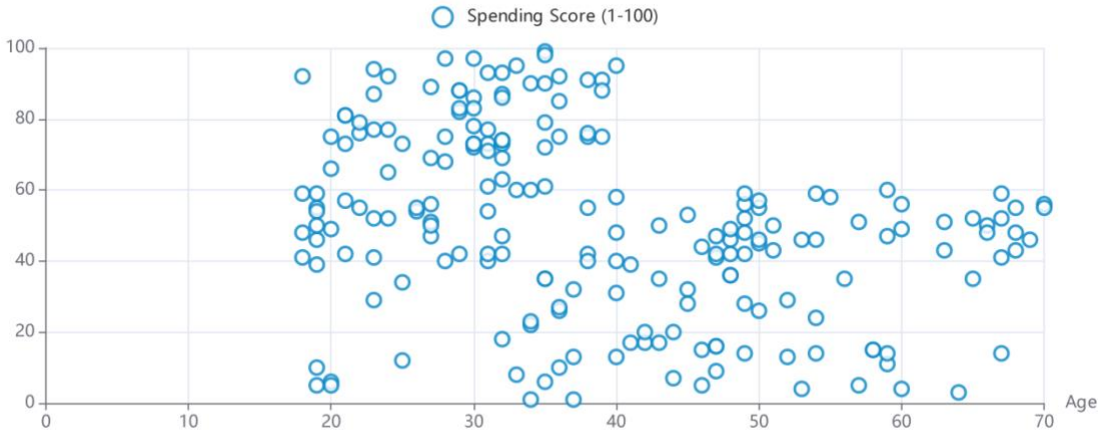
CUSTOMER SEGMENTATION USING DATASCIENCE

STEP 1 : Splitting the data set in to testing and training set

```
import pandas as pd
from sklearn.model_selection import train_test_split
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv("F:\Mall_Customers.csv")

# Split the dataset into features (X) and target variable (y)
X = df[['Genre','Age','Annual Income (k$)']]  # X is provided with the
independent variables
y = df['Spending Score (1-100)'] # y is provided with the dependent variable

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42) # here the splitting is done with 80:20 ratio
```

STEP 2 :Display the relation between the variables

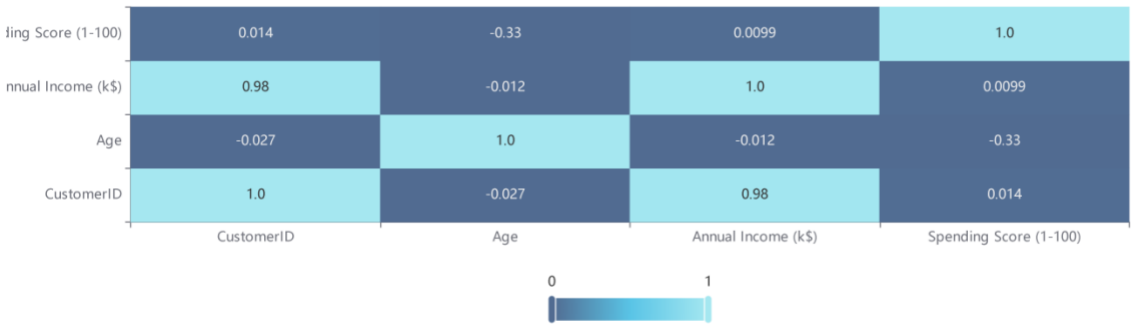
```
# Display the relation between the variables
plt.scatter(df['Age'],df['Spending Score (1-100)'])
```



```
plt.scatter(df['Annual Income (k$)'],df['Spending Score (1-100)'])
```

STEP 3 :Finding the correlation matrix and plot it in the heatmap

```
# Create a correlation matrix
correlation_matrix = df.corr()

# Create a heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='Blues', fmt='.2f',
linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```



In this project, the process of preparing and understanding the dataset involves several crucial steps. One integral step is data splitting, where the available dataset is divided into training and testing sets. This division allows for the evaluation of the model's performance on unseen data, helping to gauge its generalization capabilities. The train_test_split function from the sklearn.model_selection module is often employed for this purpose, randomly allocating a portion of the data for training and the rest for testing here it is split into 80 and 20 ratio.

Additionally, exploring the relationships between variables is vital, and one way to achieve this is by computing the correlation matrix. The correlation matrix provides a comprehensive overview of how each feature correlates with every other feature in the dataset. High positive or negative correlation values indicate strong relationships, offering insights into potential multicollinearity and guiding feature selection decisions.

Moreover, visualizing the relationships between variables can enhance the understanding of the dataset. Scatter plots serve as a valuable tool for this purpose, allowing the exploration of how two variables interact. The matplotlib library in Python is commonly used to create scatter plots, providing a visual representation of data points and trends. These plots are invaluable for identifying patterns, outliers, and potential nonlinear relationships between variables, aiding in the formulation of effective modeling strategies.